REMARKS

Applicants respectfully request reconsideration and allowance in view of the foregoing amendment and the following remarks. Applicants amend claim 1 without prejudice or disclaimer.

Rejection of Claims 1-12 Under 35 U.S.C. §112

The Office Action rejects claims 1-12 under 35 U.S.C. §112, first paragraph, as failing to comply with the written description requirement. Applicants amend claim 1 to remove the phrase "outside of a telecommunications network." Accordingly, Applicants respectfully request that the rejection be withdrawn.

Rejection of Claims 1-12 and 21-32 Under 35 U.S.C. §103(a)

The Office Action rejects claims 1-12 and 21-32 under 35 U.S.C. §103(a) as being unpatentable over Devine et al. (U.S. Publication No. 2003/0217190) ("Devine et al.") in view of Sinai et al. (U.S. Patent No. 7,143,042) ("Sinai et al."). Applicants amend claims 1 to recite augmenting the higher level representation with terminal symbols representing dynamically typed state variable assignments and comparisons associated with decision and computation shapes in the call flow. The specification provides support for this amendment at paragraph [0036]. According to paragraph [0036], dynamically typed variables can be integers, floats, and strings. Further, one generally accepted definition of dynamically typed is that type checking is done at run-time as opposed to compile time as in a statically typed approach. This generally accepted definition is provided as background and should not be interpreted as limiting the claims. Applicants submit that neither Devine et al. nor Sinai et al. teach or suggest dynamically typed state variables.

The Office Action mailed 27 October 2008 points to Devine et al., paragraph [0007] as teaching the recited limitation of "augmenting the higher level representation with terminal symbols representing state variable assignments and comparisons associated with decision and computation shapes in the call flow." However, Devine et al. teach further in paragraph [0047] that "The environment declaration block 52 declares different environmental variables and includes a series of '# include' statements that may be user defined. Additionally, global variables in this state machine may be defined in the environment descriptor block. Typically there is one environment descriptor block per state machine and multiple environment descriptors will generate an error." The global variables defined in the environment declaration block 52 are part of Devine et al.'s state machine and thus are state variables. The user defined environmental variables declare a static type which does not change throughout the program execution, such as a char, a short, an integer, a double, or a float.

Further, in paragraph [0058], Devine et al. teach that "Global variables are defined in `C` and positioned in the output file before the rest of the automatically generated code. Typically, the global variables are set appropriately before entering into the design tool and retrieved once the processing is done." First of all, Devine et al. teach that variables are defined in C. C does not provide a mechanism for dynamically typed variables because C is a static-typed language. See http://www.sitepoint.com/print/typing-versus-dynamic-typing/ , Static Typing section.

However, the C language allows for typecasting one variable type to another variable type. One simple example of a typecast is to shown by the following code snippet "result = (float) 8 / 3". In this snippet, the integer 8 is typecast to a floating point variable and divided by three, which means that the variable "result" is assigned a floating point value of 8 / 3, or 2.66666. This typecasting approach in C is different from the recited dynamically typed variables because typecasting does not convert variables, it instructs the computer to read one

variable type assuming it is another type (checked at compile time, i.e. "before entering the design tool") in contrast to a dynamically typed variable which can change types as needed and is not checked until run time. In this case, if result is not declared as a float, C cannot compile this code snippet because the types do not match. While Devine et al. teach in paragraph [0045] that other suitable programming languages can be used, such as C++, Java, Fortran and Pascal, none of these examples are dynamically typed languages. Even if a suitable dynamically typed language is used, the principles and disclosure of Devine et al. would restrict that language to use statically typed variables. For at least these reasons, Devine et al. do not teach the claim limitations as amended to recite dynamically typed state variables.

Therefore, even if combined as proposed in the Office Action, Devine et al. and Sinai et al. do not teach or suggest all the claim limitations. Applicants similarly amend claim 21. Accordingly, Applicants submit that claims 1 and 21 as well as their dependent claims are patentable over the proposed combination of Devine et al. and Sinai et al. and respectfully request that the rejection under 35 U.S.C. §103(a) be withdrawn.

CONCLUSION

Having addressed all rejections and objections, Applicants respectfully submit that the subject application is in condition for allowance and a Notice to that effect is earnestly solicited. If necessary, the Commissioner for Patents is authorized to charge or credit the **Novak, Druce & Quigg, LLP, Account No. 14-1437** for any deficiency or overpayment.

Respectfully submitted,

Date: June 3, 2009 ~~May 4, 2009~~

By: _____

Thomas M. Isaacson

Attorney for Applicants
Reg. No. 44,166
Phone: 410-286-9405
Fax No.: 410-510-1433

Correspondence Address:
Thomas A. Restaino
Reg. No. 33,444
AT&T Corp.
Room 2A-207
One AT&T Way
Bedminster, NJ 07921